Computability in the lattice of equivalence relations

Moyen, Jean-Yves; Simonsen, Jakob Grue

Published in:
Proceedings 8th Workshop on Developments in Implicit Computational Complexity and 5th Workshop on Foundational and Practical Aspects of Resource Analysis

DOI:
10.4204/EPTCS.248.8

Publication date:
2017

Document version
Publisher's PDF, also known as Version of record

Computability in the Lattice of Equivalence Relations

Jean-Yves Moyen* Jakob Grue Simonsen†

Department of Computer Science, University of Copenhagen (DIKU)
Njalsgade 128-132, 2300 Copenhagen S, Denmark
Jean-Yves.Moyen@univ-paris13.fr simonsen@diku.dk

We investigate computability in the lattice of equivalence relations on the natural numbers. We mostly investigate whether the subsets of appropriately defined subrecursive equivalence relations –for example the set of all polynomial-time decidable equivalence relations– form sublattices of the lattice.

1 Introduction

1.1 Motivation

As there are uncountably many properties of programs (because each set of programs is a property), most properties must be undecidable. A celebrated negative result shows that there are even further barriers to decidability: Rice's Theorem [10] states that every non-trivial, extensional property on programs is undecidable. An extensional property is one that depends solely on the (input-output) *function* computed by the program.

While Rice's theorem is six decades old, it still spurs new research trying to find the boundary of decidable progam properties, for example Asperti's work on complexity cliques [1]. Such properties can be studied fruitfully by viewing results such as Rice's and Asperti's as assertions about *equivalence relations*; indeed, any program property is an equivalence relation that has at most two classes: the class of programs having the property and the class of programs not having it. In this view, Rice's Theorem studies the *extensional equivalence*, $\mathfrak{R}$, where two programs are equivalent if and only if they compute the same function. Rice's theorem thus states that *no non-trivial equivalence class or union of equivalence classes in* $\mathfrak{R}$ *is decidable*.

Rather than studying individual equivalences such as $\mathfrak{R}$, it is interesting to look at the set of all equivalences between programs and at subsets of equivalences sharing certain properties. The set of all equivalence relations between programs has a very natural complete lattice structure. With this order, taking the union of some classes yields a larger equivalence relation and Rice's Theorem says that any class of any non-trivial equivalence relation in the principal filter at $\mathfrak{R}$ is undecidable.

The fact that Rice's Theorem is so neatly expressed in the lattice language hints that we can gain knowledge by studying it. One way is to look at other equivalences that the extensional one [1, 5]. Another is to look at the lattice structure itself. Since there are uncountably many equivalences, the lattice is hard to study and we want to find a good way to approximate it in a manageable way. For this reason, we study subsets of equivalences and how they interact with the lattice structure.

*Supported by the Marie Skłodowska–Curie action "Walgo", program H2020-MSCA-IF-2014, number 655222

†Partially supported by the Danish Council for Independent Research *Sapere Aude* grant "Complexity via Logic and Algebra" (COLA).

G. Bonfante, G. Moser (Eds.): 8th Workshop on Developments in Implicit Computational complExity and 5th Workshop on Foundational and Practical Aspects of Resource Analysis (DICE-FOPARA 2017)
EPTCS 248, 2017, pp. 38–46, doi:10.4204/EPTCS.248.8

The lattice-theoretic properties of $\mathrm{Equ}(S)$ are well-understood; basic facts can be gleaned from the papers [8, 11] and the textbooks [2, §8-9] [3, Sec. IV.4]. As the lattice structure of $\mathrm{Equ}(S)$ is isomorphic to $\mathrm{Equ}(T)$ for any sets S and T of identical cardinality, we may without loss of generality consider the set of equivalences over the natural numbers, $\mathrm{Equ}(\mathbb{N})$. That is, we work modulo an unspecified encoding of programs.

Because Rice's Theorem is all about computability, we are interested here in the subsets of equivalences that are defined in term of computability (*e.g.* the set of decidable equivalences) or complexity (*e.g.* the set of equivalences decidable in polynomial time). Rice's Theorem can thus be expressed as saying that the intersection of the principal filter at $\mathfrak{R}$ and the set of decidable equivalences is reduced to the trivial equivalence with one class containing everything.

Finding a subset of equivalences that is manageable but still retain most of the order-theoretic properties of the whole lattice provides a way to approximate $\mathrm{Equ}(\mathbb{N})$ (as $\mathbb{Q}$ approximates $\mathbb{R}$) and allows for a more focused and systematic study.

1.2 Equivalence relations and Lattice

We consider the set $\mathrm{Equ}(\mathbb{N})$ of all equivalences over natural numbers. If $\mathscr{E}$ is an equivalence, we write $m\mathscr{E}n$ to say that m and n are equivalent. $\mathrm{Equ}(\mathbb{N})$ is ordered by $\mathscr{E} \leq \mathscr{E}'$ iff $m\mathscr{E}n \Rightarrow m\mathscr{E}'n$. Note that this is exactly the subset ordering $\subseteq$ on $\mathbb{N} \times \mathbb{N}$. We can easily see that $\mathscr{E} \leq \mathscr{E}'$ iff each class of $\mathscr{E}'$ is the union of one or more classes of $\mathscr{E}$.

$(\mathrm{Equ}(\mathbb{N}), \leq)$ is a lattice with the following operations:

- The meet (greatest lower bound) of $\mathscr{E}$ and $\mathscr{F}$ is $\mathscr{G} = \mathscr{E} \wedge \mathscr{F}$ such that $m\mathscr{G}n$ iff $m\mathscr{E}n$ and $m\mathscr{F}n$. In this case, the classes of $\mathscr{G}$ are exactly the (non-empty) intersections of one class of $\mathscr{E}$ and one class of $\mathscr{F}$.
- The join (lowest upper bound) of $\mathscr{E}$ and $\mathscr{F}$ is $\mathscr{G} = \mathscr{E} \vee \mathscr{F}$ such that $m\mathscr{G}n$ iff there exists a finite sequence $a_1, \ldots, a_k$ such that $m\mathscr{E}a_1\mathscr{F}a_2\mathscr{E}\ldots\mathscr{F}n$.

Note that the join is much harder to express (and compute) than the meet. This will be reflected in the results. Indeed closure for meet usually boils down to closure under intersection, but closure for join is far from closure under union.

Standard results for $\mathrm{Equ}(\mathbb{N})$ were laid out by Ore [8]. It is known that $\mathrm{Equ}(\mathbb{N})$ is, among others:

- bounded with minimal element $\bot$ being the equivalence where each class is a singleton (no two different elements are equivalent) and the maximal element $\top$ being the equivalence with a single class containing every elements (any two elements are equivalents);
- complete, that is closed under arbitrary join and meet (and not only under finite ones);
- atomistic, each element is the join of atoms, where atoms are successors of $\bot$, that is equivalences with one class containing two elements and the other are singletons;
- relatively complemented (hence complemented), for each $\mathscr{E}$, there exists $\mathscr{F}$ such that $\mathscr{E} \vee \mathscr{F} = \top$ and $\mathscr{E} \wedge \mathscr{F} = \bot$; non $\top$ or $\bot$ equivalences have infinitely many complements, most have uncountably many.

An equivalence relation $\mathscr{E}$ where exactly one class is not a singleton is called *singular*.

Lemma 1.1 (Complements to singular equivalence relations)**.** *Let $\mathscr{E}$ be a singular equivalence relation with non-singleton class E. $\mathscr{F}$ is one of its complement iff each class of $\mathscr{F}$ contains exactly one element of E.*

To avoid confusion with meet and join, we note && and || the logical conjunction and disjunction. We note $\mathtt{n}$ the binary representation of n.

1.3 Computability and complexity

We refer to standard textbooks convering computability and complexity theory (*e.g.*, [4]).

Unless otherwise stated, all Turing Machines are multi-tape machines with two designated read-only input tapes, one write-only output tape, and any number of work tapes. Machines are deterministic unless otherwise stated.

Equivalences relations are then classified in the natural way with respect to the Turing Machine deciding them. That is, for example, "polynomial time relations" or "recursively enumerable relations" are defined in the straightforward way.

1.4 Results

The paper considers three broad categories: automatic, subrecursive, and arithmetical equivalence relations and how they interact with the basic properties of the entire lattice of equivalence relations, namely meet, join and complements. The results are summarised here.

	Finite		Arithmetical		Arbitrary	
	$\wedge$	$\vee$	$\wedge$	$\vee$	$\wedge/\vee$	complements
Automatic	Yes	Yes	No	Yes	No/Yes	N/A
Subrecursive	Yes	No†	?	No†	No	$\geq$ PSPACE
Σ^0_k	Yes	Yes	No	Yes	No	No
Π^0_k	Yes	No	Yes	No	No	?
Δ^0_k	Yes	No	No	No	No	Yes

†: for LOGSPACE or larger classes.

Note that a "Yes" entry does not imply any results concerning actual computation of the operation –it merely implies that the subset of equivalence relations is closed under the operation considered– but the actual computation involved in the operation may require resources beyond those implied by the subset. If we consider a subset S of equivalences and an operation on the equivalences, there are four possibilities:

1. S is not closed under this operation, that is there exist elements in S such that, when applying the operation to them we can obtain an element not in S. *This is the canonical meaning of "No" in the above table.*
2. S is closed under the operation (i.e., there is a "Yes" in the above table), and either
   (a) the operation is computable within the resource requirements defining the class, e.g. polytime computable for the set of polytime decidable equivalence relations.
   (b) the operation is computable, but not within the resource requirements defining the class.
   (c) the operation is not computable.

2 Automatic equivalence relations

There are several "natural" ways to define equivalence relations decidable by finite automata. The one we consider here uses a single, two-way input tape with two inputs separated by a special symbol; another natural variation would have the two inputs on two separate tapes, or a single tape but two heads. Unlike Turing machines, the class of sets decidable by finite automata properly increases with the number of input tapes and tape heads; it is therefore quite conceivable that simple variations of the class we consider would have different properties.

We consider two-way finite automata that have only a single tape where both inputs are encoded. We believe the most natural form to be $a\square b$ where $a,b \in \{0,1\}^+$ and $\square$ is a separator symbol ("blank"). We denote $\mathrm{Equ}(\mathbb{N})_{\mathrm{AUT}}$ the set of *automatic* equivalence relations, that is such that the language $\{\mathtt{m}\square\mathtt{n} : m \mathscr{E} n\}$ is accepted by a two-way non-deterministic finite automaton. By standard results, this language is thus regular. That is, 2-way Non-Deterministic Automata recognise exactly the same languages as 1-way Deterministic Automata.

Once such an automaton is fixed, for any integer m we note q_m the state reached after reading $\mathtt{m}\square$.

Proposition 2.1. *Let $\mathscr{E} \in Equ(\mathbb{N})_{\mathrm{AUT}}$. It has finitely many equivalence classes.*

The proof uses an argument taken from the Myhill-Nerode Theorem [7].

Sketch of proof. The automaton must accept $\mathtt{m}\square\mathtt{m}$ by reflexivity of equivalences. Hence, by determinism, if $q_n = q_m$, the automaton must also accept $\mathtt{n}\square\mathtt{m}$ and $n \mathscr{E} m$. Thus, there can be at most as many classes as states. □

Note that Proposition 2.1 implies that some very simple equivalence relations are not elements of $\mathrm{Equ}(\mathbb{N})_{\mathrm{AUT}}$; for example, $\bot \notin \mathrm{Equ}(\mathbb{N})_{\mathrm{AUT}}$ (that is, $\{\mathtt{n}\square\mathtt{n}\}$ is not a regular language, a well-known fact).

Since the number of classes in any equivalence of $\mathrm{Equ}(\mathbb{N})_{\mathrm{AUT}}$ is finite but unbounded, given such an equivalence, it is always possible to find a class with several elements and create a new equivalence by taking one element out of this class and making a new singleton class. This new equivalence is smaller than the initial one and is still single-tape automatic. Hence, $\mathrm{Equ}(\mathbb{N})_{\mathrm{AUT}}$ is neither bounded, nor complete.

Moreover, let $\mathscr{E}$ and $\mathscr{F}$ be two equivalences, each with finitely many classes $E_1,\dots,E_m$ and $F_1,\dots,F_n$. Since the classes of $\mathscr{E} \wedge \mathscr{F}$ are exactly the non-empty $E_i \bigcap F_j$, there are at most $n \times m$ such classes, that is, a finite number, thus it cannot be $\bot$ and $\mathscr{F}$ cannot be a complement to $\mathscr{E}$.

Hence, no complement of a single-tape automatic equivalence is single-tape automatic.

Theorem 2.2. *$Equ(\mathbb{N})_{\mathrm{AUT}}$ is a lattice.*

Sketch of proof. The language for the meet is the intersection of both languages and regular languages are closed under intersection.

In order to build an automaton recognising the join, we first start by finding representatives of each class of both equivalences : $m_1,\dots,m_e$ and $n_1,\dots,n_f$; this is doable by standard search in minimal DFA. Next we precompute which m_i and n_j are related in $\mathscr{E} \vee \mathscr{F}$; this is doable because the finite number of classes implies a bounded length for the chain when unfolding the join definition.

Lastly, we can build an NFA with ε-transitions that first check which unique m_i is such that $m \mathscr{E} m_i$ (ε-transitions to e copies of the first automaton); and then check if n is such that $n \mathscr{F} n_j$ for one of the n_j related to m_i (correct number of ε-transitions to copies of the second automata). □

Thus, not only is $\mathrm{Equ}(\mathbb{N})_{\mathrm{AUT}}$ a lattice, but both the meets and the joins are effectively computable. Computing these, however, cannot be performed by a DFA (this is case 2b of the discussion after the results Table).

Proposition 2.3. *Let $k \geq 1$. The meet of a Σ^0_k set of automatic equivalence relations is not necessarily automatic.*

Proof. Let, for each $i \geq 1$, $\mathscr{E}_i$ be the automatic equivalence relation containing the two classes $\{i\}$ and $\mathbb{N} \setminus \{i\}$. Then, $\bigwedge \mathscr{E}_i = \bot$, which is not automatic. □

Lemma 2.4. *$Equ(\mathbb{N})_{\mathrm{AUT}}$ is upward closed.*

Proof. Let $\mathscr{E} \in \mathrm{Equ}(\mathbb{N})_{\mathrm{AUT}}$ and $\mathscr{E} \leq \mathscr{F}$. By definition, the classes of $\mathscr{F}$ are unions of classes of $\mathscr{E}$ and because there are only finitely many classes in $\mathscr{E}$, these are *finite* unions. Thus, building a DFA for $\mathscr{F}$ is possible. □

Proposition 2.5. *Let $A \subseteq Equ(\mathbb{N})_{\mathrm{AUT}}$ be non-empty. Then, $\bigvee A \in Equ(\mathbb{N})_{\mathrm{AUT}}$.*

Proof. Because $\mathscr{E} \leq \bigvee A$ for any $\mathscr{E} \in A$. □

As mentioned at the beginning of the Section, the expressive power of automata varies with the number of tapes or heads. Hence, small variations of the model can drastically change the property of the corresponding subset of equivalences. For example, two-tape automata can test whether both tape contain the same word and thus decide $\perp$.

Even the choice of representation of inputs may affect the expressive power. For example, instead of sequencing the inputs ($\mathtt{m}\Box\mathtt{n}$), it is possible to interleave them ($m_1n_1m_2n_2\ldots m_in_i\ldots\Box n_k$ if n is longer than m). This representation allows an automaton to decide $\perp$ (note that the Myhill-Nerode argument does not work in this case).

Moreover, when working with multi-tape automata, the question of synchronicity arises: should the tapes be "at the same position" on each tape, or can the heads move independently?

Thus, we have only studied one particular class of automata with one particular way of representing equivalences. $\mathrm{Equ}(\mathbb{N})_{\mathrm{AUT}}$ hence enjoys properties that other classes of "automatic" relation may or may not have. While the class of single-tape, single-head DFAs studied above is the simplest kind of finite automaton –hence most apt to study first– it remains to perform a systematic study of equivalence relations decidable by other, more particular, kinds of automata.

3 Subrecursive equivalence relations

We now treat classes of equivalence relations decidable within bounds on their resources. The definition of such classes are simply the standard definitions of computational complexity theory using Turing machines with two input tapes. The sets of equivalence relations on $\mathbb{N}$ consisting of LOGSPACE-, PTIME-, PSPACE-, EXPTIME-, primitive recursive, ... decidable equivalence relations are denoted by $\mathrm{Equ}(\mathbb{N})_{\mathrm{LOGSPACE}}$, $\mathrm{Equ}(\mathbb{N})_{\mathrm{PTIME}}$, $\mathrm{Equ}(\mathbb{N})_{\mathrm{PSPACE}}$, $\mathrm{Equ}(\mathbb{N})_{\mathrm{EXPTIME}}$, $\mathrm{Equ}(\mathbb{N})_{\mathrm{p.r.}}$, ... The notation is extended in straightforward ways and we collectively call these "subrecursive" sets of equivalences.

Lemma 3.1. *The subrecursive sets of equivalences are closed under finite meet.*

Proof. $m(\mathscr{E} \wedge \mathscr{F})n$ iff $m\mathscr{E}n \,\&\&\, m\mathscr{F}n$, and the subrecursive classes are closed under &&. □

Proposition 3.2. *There exist two equivalence relations, decidable in logarithmic space, whose join is undecidable.*

Thus, the subrecursive sets of equivalences are not closed under join and are not sublattices.

Sketch of proof. Let M be a deterministic Turing Machine. We define the *clocked one-step* relation by $(\mathtt{n},\mathtt{c}) \to (\mathtt{n}',\mathtt{c}')$ iff $n' = n+1$ and $\mathtt{c}'$ is the (representation of the) configuration resulting from executing one step M from $\mathtt{c}$; or if $\mathtt{c}$ is a final configuration and $(\mathtt{n}',\mathtt{c}') = (0,0)$. $\to$ is decidable in logarithmic space by standard techniques.

We now define $\to_{\text{even}}$ (resp. $\to_{\text{odd}}$) as the restriction of $\to$ for even (resp. odd) n; and $\approx_{\text{even}}$ (resp. $\approx_{\text{odd}}$) as the reflexive, transitive, symmetric closure of $\to_{\text{even}}$ (resp. $\to_{\text{odd}}$). Because of the parity restriction, it is not possible to have $(\mathtt{n}_0,\mathtt{c}_0) \to_{\text{even}} (\mathtt{n}_1,\mathtt{c}_1) \to_{\text{even}} (\mathtt{n}_2,\mathtt{c}_2)$ and thus $\approx_{\text{even}}$ is also decidable in logarithmic space.

Let $\approx = (\approx_{\text{even}} \vee \approx_{\text{odd}})$. The computation starting at c terminates iff $(\mathtt{n},\mathtt{c}) \approx (0,0)$. Hence, $\approx$ is not decidable. □

Theorem 3.3. *Let A be a (deterministic) subrecursive set of equivalences larger than $Equ(\mathbb{N})_{\text{PSPACE}}$ (included) and $\mathscr{E} \in A$. There is at least one complement to $\mathscr{E}$ in A.*

Sketch of proof. Let $\mathscr{F}$ be the singular equivalence whose non-singleton class, F, contains exactly the least element of each class of $\mathscr{E}$. It is a complement to $\mathscr{E}$ by Lemma 1.1. Because the deterministic subrecursive sets are closed under complement, $\overline{\mathscr{E}} \in A$.

To decide if $m\mathscr{F}n$, proceed as follows: (i) if $\mathtt{m} = \mathtt{n}$, accept; (ii) if there exists $k < m$ with $\mathtt{k}\mathscr{E}\mathtt{m}$, reject; (iii) if there exists $k' < n$ with $\mathtt{k}'\mathscr{E}\mathtt{n}$, reject; (iv) if not rejected yet, accept.

Step (ii) is done by checking if for all $k < m$, $k\overline{\mathscr{E}}m$ and requires $O(|\mathtt{m}|)$ working space for writing k, hence at least linear space. It also loops over $m = 2^{|\mathtt{m}|}$ values and thus needs exponential time. □

It is tempting to conjecture that there are equivalences in $\text{Equ}(\mathbb{N})_{\text{PTIME}}$ with no complement in $\text{Equ}(\mathbb{N})_{\text{PTIME}}$. However if that were the case, we would have $\text{Equ}(\mathbb{N})_{\text{PTIME}} \neq \text{Equ}(\mathbb{N})_{\text{PSPACE}}$, and by using a polynomial-time pairing function $\mathbb{N}^2 \longrightarrow \mathbb{N}$, existence of $\mathscr{P} \in \text{Equ}(\mathbb{N})_{\text{PSPACE}} \setminus \text{Equ}(\mathbb{N})_{\text{PTIME}}$ entails existence of a set in PSPACE $\setminus$ PTIME, and hence PSPACE $\neq$ PTIME; hence, the conjecture will be exceedingly hard to prove. On the other hand, It is not clear that equality of $\text{Equ}(\mathbb{N})_{\text{PTIME}}$ and $\text{Equ}(\mathbb{N})_{\text{PSPACE}}$ would entail PSPACE = P.

4 Arithmetical equivalence relations

We now investigate closure properties of sets of arithmetical equivalence relations. Closure under finite meet is immediate as the arithmetical sets are also closed under finite intersections.

Lemma 4.1. *For every $k \geq 1$, the set of equivalence relations in Σ^0_k is closed under finite join.*

The join of any two Π^0_k equivalence relations is in Σ^0_{k+1}.

Proof. Let x and y be two integers. By definition, $x(\mathscr{E} \vee \mathscr{F})y$ iff there exists $a_1,\ldots,a_n$ such that $x\mathscr{E}a_1\mathscr{F}\ldots\mathscr{E}a_n\mathscr{F}y$. That is $\exists n,a_1,\ldots,a_n.x\mathscr{E}a_1 \,\&\&\, a_1\mathscr{F}a_2 \,\&\&\, \ldots \,\&\&\, a_n\mathscr{F}y$, which is Σ^0_k (resp. Σ^0_{k+1}) if $\mathscr{E}$ and $\mathscr{F}$ are both Σ^0_k (resp. Π^0_k). □

Proposition 4.2. *For any $k \geq 1$, the set of equivalence relations in Δ^0_k is not closed under finite join.*

Idea of proof. The proof is essentially the same as for Proposition 3.2. The "hard" problem cannot stay the classical halting problem and must be replaced by the halting problem for machines with oracle in Δ^0_k. □

Proposition 4.3. *For every $n \geq 1$, the set of equivalence relations in Π^0_n is not closed under finite join.*

Proof. Because $\Delta^0_k = \Sigma^0_k \cap \Pi^0_k$ is not, but Σ^0_k is. □

Theorem 4.4. *For every $k \geq 1$, there are equivalence relations in Σ^0_k none of whose complements are in Σ^0_k.*

Sketch of proof. Let E be a Σ^0_k set whose complement is not Σ^0_k, and $\mathscr{E}$ be the singular equivalence with non-singleton class E. Let $\mathscr{F}$ be a complement to $\mathscr{E}$, each of its class intersects E in exactly one point, hence $x \in \overline{E}$ iff $\exists e.e \neq x \,\&\&\, e \in E \,\&\&\, e \mathscr{F} x$, and $\mathscr{F}$ cannot be Σ^0_k. □

Theorem 4.5. *Let $k \geq 0$. Every equivalence relation in Δ^0_k has at least one complement in Δ^0_k.*

The proof is essentially the same as for Theorem 3.3.

5 Infinite meet and join

Recall that the upper set $\uparrow \{n\}$ contains n and all the elements greater than n. We say that an equivalence is *small* if it has finitely many classes. Note that small singular equivalence relations have finitely many singleton classes, whose largest element is N and the non-singleton class is thus the union of a finite set and the upper set $\uparrow \{N+1\}$.

Proposition 5.1. *Let $I \subseteq \mathbb{N}$, there is a set of small singular equivalences whose meet is singular with non-singleton class I. Thus, none of the subrecursive or arithmetical sets of equivalences are closed under arbitrary meet.*

Sketch of proof. Let $I \subseteq \mathbb{N}$. Let $(f_i)_{i\in\mathbb{N}}$ be a strictly increasing sequence and let $F_i = I \bigcap [0; f_i[$ and $F_i^+ = (F_i \bigcup \uparrow \{f_i\})$. Note that $I = \bigcap_i F_i^+$. Let $\mathscr{F}_i$ be the small singular equivalence with non-singleton class F_i^+.

$\bigwedge \mathscr{F}_i$ is the singular equivalence whose non-singleton class is I. □

Proposition 5.2. *Let $I \subseteq \mathbb{N}$. There exists a set of atoms whose join is singular with non-singleton class I.*

Thus, none of the arithmetical sets of equivalences are closed under arbitrary join.

Proof. Because the lattice is atomistic. Atoms (singular equivalences whose non-singleton set has only two elements) are decidable in zero space and linear time (by encoding the two elements in the states of the machine). □

The previous results assume that the set of equivalence relations may be chosen arbitrarily. Especially, it is based on a set I of arbitrary difficulty. This means that the results boil down to the fact that there are uncountably many such I, while there are only countably many arithmetical relations.

We now look at what happens if the set of relations itself is to have some bound. Specifically, we are concerned with the meet and join of a Σ^0_k set of equivalences. Σ^0_k (and especially, recursively enumerable) is a sensible bound: It means that one can enumerate each of the equivalences (with a proper oracle, or none for recursively enumerable) until all the needed ones have been found. On the other hand, a Π^0_k set of equivalences would mean that one can enumerate (with oracle) the complement to this set, which is less practical.

Proposition 5.3. *For all $k \geq 1$, the join of a Σ^0_k set of Σ^0_k equivalences is a Σ^0_k equivalence.*

Proof. Let $\{\Phi_i : i \in I\}$ be a Σ^0_k set of Σ^0_k equivalences, that is I is a Σ^0_k set of integers and each Φ_i is a Σ^0_k equivalence. Let $\mathscr{E} = \bigvee \Phi_i$. By definition, we have $x \mathscr{E} y$ if and only if

$$\exists n, a_1, \ldots, a_n, x_1, \ldots, x_n. \quad a_1 \in I \,\&\&\, \ldots \,\&\&\, a_n \in I \quad \&\& x \Phi_{a_1} x_1 \,\&\&\, x_1 \Phi_{a_2} x_2 \,\&\&\, \ldots \,\&\&\, x_n \Phi_{a_n} y$$

□

Note that because $i \leq j$ implies $\Sigma^0_i \subset \Sigma^0_j$, we immediately have that the join of a Σ^0_i set of Σ^0_j equivalences is Σ^0_j. In particular, the join of a recursively enumerable set of Σ^0_j equivalences is a Σ^0_j equivalence relation.

Proposition 5.4. *For all positive integers n, there exists a Σ^0_n set of Σ^0_n equivalence relations whose meet is not Σ^0_n.*

Sketch of proof. We consider an encoding of formulae into numbers and let I be the set of encoding of Σ^0_k formulae, it is decidable if the encoding allows to count quantifiers. Let $A_k = \{ i \in I \,:\, \Phi_i(k) \}$ be the set of Σ^0_n equivalence accepting k, it is Σ^0_n. Let $\mathscr{A}_k$ be the singular equivalence whose non-singleton set is A_k, it is a Σ^0_k equivalence.

Now, $\bigwedge \mathscr{A}_k$ is singular with non-singleton class the (encoding of) Σ^0_k tautologies. This is a Π^0_{k+1}-complete set. □

Proposition 5.5. *For all k, the meet of a Σ^0_k set of Π^0_k equivalence relations is Π^0_k.*

Proof. Let $\{ \Phi_i \,:\, i \in I \}$ be a Σ^0_k set of Π^0_k equivalences and $\mathscr{E} = \bigwedge \Phi_i$. By definition, $x \mathscr{E} y$ iff $\forall i, i \in I \Rightarrow x \Phi_i y$, which is equivalent to $\forall i, i \notin I \,||\, x \Phi_i y$, a Π^0_k formula. □

Here also, by inclusion of the hierarchy, $i \leq j$ implies that any Σ^0_i set of Π^0_j equivalence relations has a Π^0_j meet. Especially, the set of Π^0_k equivalence relations is closed under recursively enumerable meet.

Proposition 5.6. *There is an r.e. set of elements of $Equ(\mathbb{N})_{\text{LogSpace}}$ whose meet is not decidable.*

Sketch of proof. Let $\mathscr{E}_n$ be the singular equivalence whose non-singleton set is the (representation of) Turing Machines that *do not* halt in n or less step. It is a LogSpace equivalence by clever encoding of TMs (see, *e.g.*, [9, Ch. 3]), the main trick here being that we only need to simulate a *fixed* number of steps (n) and this requires a *fixed* amount of extra space (plus logarithmic overhead for the simulation). However, $\bigwedge \mathscr{E}_n$ is the singular equivalence whose non-singleton class is the Turing Machines who never halt and is thus not decidable. □

Corollary 5.7. *The subrecursive sets of equivalences are not closed under arithmetical meets.*
The sets of Π^0_k equivalences are not closed under arithmetical meets.

For the second point, the proof must be adapted using the halting problem for the correct class of oracle machines.

6 Conclusion and Future Works

Out of the three classes of equivalences that we have considered, the automatic ones seem too few to be of interest; the subrecursive ones do not form a sublattice, making them bad candidates for studying the lattice structure; but the arithmetical ones seem more interesting. Notably, the Σ^0_k sets of equivalences do keep the lattice structure and especially $\text{Equ}(\mathbb{N})_{\text{r.e.}}$ is worth more efforts.

In addition to being the smallest Σ^0_k set of equivalences, $\text{Equ}(\mathbb{N})_{\text{r.e.}}$ is also linked back to the starting point via the Rice-Shapiro's Theorem [6, 12]: while $\mathfrak{R} \notin \text{Equ}(\mathbb{N})_{\text{r.e.}}$, we already know something about equivalences in the intersection of $\text{Equ}(\mathbb{N})_{\text{r.e.}}$ and the principal filter at $\mathfrak{R}$.

Since there are equivalences in $\text{Equ}(\mathbb{N})_{\text{r.e.}}$ with no complements in it, we do not get all the basic lattice property (the sublattice is not complemented). Thus, we may still want to find other sublattices defined by other criteria.

Other questions raised by this work include a more systematic study of the subsets decided by various kind of automata; and trying to build, for each equivalence in $\mathrm{Equ}(\mathbb{N})_{\mathrm{PTIME}}$, a complement in $\mathrm{Equ}(\mathbb{N})_{\mathrm{PTIME}}$.

References

[1] Andrea Asperti (2008): *The Intensional Content of Rice's Theorem*. In: *Proceedings of the 35th Annual ACM SIGPLAN - SIGACT Symposium on Principles of Programming Languages (POPL 2008)*, doi:10.1145/1328438.1328455.

[2] Garrett Birkhoff (1940): *Lattice Theory*. *Colloquium Publications* 25, American Mathematical Society.

[3] George Grätzer (2003): *General Lattice Theory*, second edition. Birkhäuser.

[4] Neil D. Jones (1997): *Computability and Complexity, from a Programming Perspective*. MIT press.

[5] J.-Y. Moyen & J. G. Simonsen (2016): *More intensional versions of Rice's Theorem*. In D. Mazza, editor: *Developments in Implicit Computational Complexity, DICE'16*, Eindhoven, Netherlands.

[6] John R. Myhill & John Cedric Shepherdson (1955): *Effective operations on partial recursive functions*. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik* 1, pp. 310–317, doi:10.1002/malq.19550010407.

[7] Anil Nerode (1958): *Linear Automaton Transformations*. *Proceedings of the American Mathematical Society* 9(4), pp. pp. 541–544, doi:10.1090/S0002-9939-1958-0135681-9. Available at http://www.jstor.org/stable/2033204.

[8] Øystein Ore (1942): *Theory of equivalence relations*. *Duke Mathematical Journal* 9(3), pp. 573–627, doi:10.1215/S0012-7094-42-00942-6.

[9] Christos H. Papadimitriou (1994): *Computational Complexity*. Addison-Wesley.

[10] Henry Gordon Rice (1953): *Classes of Recursively Enumerable Sets and Their Decision Problems*. *Transactions of the American Mathematical Society* 74, pp. 358–366, doi:10.1090/S0002-9947-1953-0053041-6.

[11] Ivan Rival & Miriam Stanford (1992): *Algebraic Aspects of Partition Lattices*. In Neil White, editor: *Matroid Applications*, *Encyclopedia of Mathematics and its Applications* 40, Cambridge University Press, pp. 106–122, doi:10.1017/CBO9780511662041.006.

[12] Normann Shapiro (1956): *Degrees of computability*. *Transactions of the AMS* 82, pp. 281–299, doi:10.1090/S0002-9947-1956-0085187-3.